Relational Algebra

Relational Algebra

Relational Algebra is a procedural data manipulation language (**DML**).

It specifies operations on relations to define new relations:

Unary Relational Operations: Select, Project

Operations from Set Theory: Union, Intersection, Difference, Cartesian Product

Binary Relational Operations: Join, Divide.

SELECT

- The SELECT operation is used to choose a *subset* of the tuples (rows) from a relation that satisfies a **selection condition**, denoted by:

$$\sigma_{<selection\ condition>}(R)$$

- The Boolean expression specified in <selection condition> is made up of a number of **selection clauses** of the form

  <attribute name> <comparison op> <constant value>

  or

  <attribute name> <comparison op> <attribute name>

- <comparison op> is normally one of the comparison operators {=, <, ≤, >, ≥, ≠}

- Selection clauses can be connected by the standard Boolean operators <u>*and*</u>, <u>*or*</u>, and <u>*not*</u> to form a general selection condition.

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE:

Department	Degree
Psychology	Ph.D.
Comp.Sci.	Ph.D.
Comp.Sci.	M.Sc.
Psychology	M.Sc.

ENROLMENT:

Enrolment#	Supervisee	Supervisor	Department	Degree
1	1	2	Psychology	Ph.D.
2	3	1	Comp.Sci.	Ph.D.
3	4	1	Comp.Sci.	M.Sc.
4	5	1	Comp.Sci.	M.Sc.

ENROLMENT:

Enrolment#	Supervisee	Supervisor	Department	Degree
1	1	2	Psychology	Ph.D.
2	3	1	Comp.Sci.	Ph.D.
3	4	1	Comp.Sci.	M.Sc.
4	5	1	Comp.Sci.	M.Sc.

Q: Select the enrolment records for the students whose supervisor is Person 1

$$\sigma_{(Supervisor=1)}(ENROLMENT)$$

Enrolment#	Supervisee	Supervisor	Department	Degree
2	3	1	Comp.Sci	Ph.D.
3	4	1	Comp.Sci	M.Sc.
4	5	1	Comp.Sci	M.Sc.

ENROLMENT:

Enrolment#	Supervisee	Supervisor	Department	Degree
1	1	2	Psychology	Ph.D.
2	3	1	Comp.Sci.	Ph.D.
3	4	1	Comp.Sci.	M.Sc.
4	5	1	Comp.Sci.	M.Sc.

Q: Select the enrolment records for Person 1's non-Ph.D. students

$$\sigma_{(Supervisor=1\ AND\ Degree\neq' Ph.D.')}(ENROLMENT)$$

$$\sigma_{(Supervisor=1\ AND\ NOT\ Degree=' Ph.D.')}(ENROLMENT)$$

Enrolment#	Supervisee	Supervisor	Department	Degree
3	4	1	Comp.Sci	M.Sc.
4	5	1	Comp.Sci	M.Sc.

Properties of SELECT

- Commutative:

$$\sigma_{<cond1>}(\sigma_{<cond2>}(R)) = \sigma_{<cond2>}(\sigma_{<cond1>}(R))$$

- Consecutive selects can be combined:

$$\sigma_{<cond1>}(\sigma_{<cond2>}(R)) = \sigma_{<cond1>\ AND\ <cond2>}(R))$$

PROJECT

- The PROJECT operation is used to project a subset of the attributes (column) of a relation, denoted by:

$$\pi_{<attribute\ list>}(R)$$

- The result of the PROJECT operation has only the attributes specified in <attribute list> in the *same order as they appear in the list*. Hence, it's **degree** is equal to the number of attributes in <attribute list>.

- The PROJECT operation *removes any duplicate tuples*, so the result of the PORJECT operation is a set of distinct tuples, and this is known as **duplicate elimination**.

ENROLMENT:

Enrolment#	Supervisee	Supervisor	Department	Degree
1	1	2	Psychology	Ph.D.
2	3	1	Comp.Sci.	Ph.D.
3	4	1	Comp.Sci.	M.Sc.
4	5	1	Comp.Sci.	M.Sc.

Q: Find departments and degree requirements for the courses that students enroll.

$$\pi_{\{department, degree\}}(ENROLMENT)$$

Department	Degree
Psychology	Ph.D.
Comp.Sci	Ph.D.
Comp.Sci	M.Sc.

Properties of PROJECT

- if <list2> contains all the attributes in <list1> then

$$\pi_{<list1>}(\pi_{<list2>}(R)) = \pi_{<list1>}(R)$$

else

The operation is not well defined.

- commutes with selection:

$$\pi_X(\sigma_B(R)) = \sigma_B(\pi_X(R)) \ (?)$$

Commutes follows if and only if the attribute names used in SELECT is a subset of the attribute list in PROJECT

Check the example below:

$$\pi_{\{degree\}}(\sigma_{(Department='Psychology')}(ENROLMENT)) =$$

Degree
Ph.D.

$$\sigma_{(Department='Psychology')}(\pi_{\{degree\}}(ENROLMENT)) =$$ Error as SELECT cannot find Degree

Questions:

1) $\pi\ (R \cup S)) = \pi\ (R) \cup \pi\ (S)$?

2) $\pi\ (R \cap S)) = \pi\ (R) \cap \pi\ (S)$?

Answer:

2) $\pi\ (R \cap S)) \neq \pi\ (R) \cap \pi\ (S)$

Example:

R = (Animal, Cat), S = (Animal, Dog)

π: project on the first column

$\pi\ (R \cap S)) = \{\}$

$\pi\ (R) \cap \pi\ (S) = \{\text{Animal}\}$

UNION

- UNION is a relation that includes all tuples that are either in the left relation or in the right relation or in both relations, denoted by

$$R \cup S = \{t : t \in R \, or \, t \in S\}$$

- Note: Union requires R and S to be **union compatible**:
  that there is a 1-1 correspondence between their attributes,
  in which corresponding attributes are over the same domain

Example:

$$R1 \leftarrow \sigma_{(Supervisor=2)} (ENROLMENT)$$

$$R2 \leftarrow \sigma_{(Name='M.Sc')} (ENROLMENT)$$

$$R1 \cup R2 =$$

Enrolment#	Supervisee	Supervisor	Department	Name
1	1	2	Psych.	Ph.D.
3	4	1	Comp.Sci	M.Sc
4	5	1	Comp.Sci	M.Sc

Example: $STUDENT \cup RESEARCHER =$

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledhill
5	Ms B.K.Lee
2	Dr R.G.Wilkinson

INTERSECTION

- INTERSECTION is a relation that includes all tuples that are in both relations, denoted by

$$R \cap S = \{t : t \in R \, and \, t \in S\}$$

- Example:

$$R_1 \leftarrow \sigma_{(Supervisor=1)}(ENROLMENT)$$
$$R_2 \leftarrow \sigma_{(Degree='Ph.D.')}(ENROLMENT)$$
$$R_1 \cap R_2 =$$

Enrolment#	Supervisee	Supervisor	Department	Name
2	3	1	Comp.Sci.	Ph.D.

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

Example: STUDENT ∩ RESEARCHER =

Person#	Name
1	Dr C.C. Chen

DIFFERENCE

- SET DIFFERENCE is a relation that includes all tuples that are in the left relation but not in the right relation, denoted by

$$R - S = \{t : t \in R \, and \, t \notin S\}$$

- Example: STUDENT – RESEARCHRER =

Person#	Name
3	Ms K. Juliff
4	Ms J. Gledhill
5	Ms B.K. Lee

CARTESIAN PRODUCT

$$R \times S = \{t_1 || t_2 : t_1 \in R \, and \, t_2 \in S\}$$

- Where $t_1 || t_2$ indicates the concatenation of tuples.

- Example: STUDENT X RESEARCHRER =

E'ment#	S'ee	S'or	D'ment	Degree	Person#	Name
1	1	2	Psych.	Ph.D.	1	Dr C.C. Chen
1	1	2	Psych.	Ph.D.	2	Dr R.G.Wilkinson
2	3	1	Comp.Sci	Ph.D.	1	Dr C.C. Chen
2	3	1	Comp.Sci	Ph.D.	2	Dr R.G.Wilkinson
3	4	1	Comp.Sci	M.Sc.	1	Dr C.C. Chen
3	4	1	Comp.Sci	M.Sc.	2	Dr R.G.Wilkinson
4	5	1	Comp.Sci	M.Sc.	1	Dr C.C. Chen
4	5	1	Comp.Sci	M.Sc.	2	Dr R.G.Wilkinson

More useful is:

$$R_1 \leftarrow ENROLMENT \times RESEARCHER$$

$$\sigma_{(Supervisor=Person\#)}(R_1) =$$

E'ment#	S'ee	S'or	D'ment	E'ment. Name	Person#	R'cher. Name
1	1	2	Psych.	Ph.D.	2	Dr R.G.Wilkinson
2	3	1	Comp.Sci.	Ph.D.	1	Dr C.C. Chen
3	4	1	Comp.Sci.	M.Sc.	1	Dr C.C. Chen
4	5	1	Comp.Sci.	M.Sc.	1	Dr C.C. Chen

Or even better:

$$R_1 \leftarrow ENROLMENT \times RESEARCHER$$

$$R_2 \leftarrow \sigma_{(Supervisor=Person\#)}(R_1)$$

$$\pi_{\{E'ment\#, S'ee, S'or, Name, D'ment, Degree\}}(R_2) =$$

E'ment#	S'ee	S'or	Name	D'ment	Degree
1	1	2	Dr R.G.Wilkinson	Psych.	Ph.D.
2	3	1	Dr C.C. Chen	Comp.Sci.	Ph.D.
3	4	1	Dr C.C. Chen	Comp.Sci.	M.Sc.
4	5	1	Dr C.C. Chen	Comp.Sci.	M.Sc.

The last of these is also known as natural join, the next to last is equi-join.

JOIN

- JOIN is used to combine related tuples from two relations into single "longer" tuples.
- **Theta-join**

$$R \bowtie_{<join\ condition>} S = \{t_1 || t_2 : t_1 \in R\ and\ t_2 \in S\ and\ <join\ condition>\}$$

- A general join condition is of the form:

<condition> **AND** <condition> **AND** … **AND** <condition>

- where each condition is of the form $A_i \theta B_j$, in which A_i is an attribute of R, B_j is an attribute of S, A_i and B_j have the same domain, and θ is a comparison operator. A JOIN operation with such a general join condition is called a **THETA JOIN**.

JOIN: Equi-join

EQUI-JOIN is a theta-join where the only comparison operator used is "=".

Example:

$$ENROLMENT \bowtie_{(Supervisor=Person\#)} RESEARCHER$$

JOIN: Natural join

NATURAL JOIN is an equi-join which requires that the two join attributes (or each pair of join attributes) have the same name in both relations.

Example:

$$ENROLMENT \bowtie_{(Supervisor),(Person\#)} RESEARCHER$$

Question: If two relations have no join attributes, how do you define the join result? Why?

$$R\ (A, B) \bowtie S\ (B, C) \bowtie T\ (C, D)$$

Notes:

1. In a natural join, there may be several pairs of

join attributes.

Example:

COURSE		
Department	Name	By
Comp.Sci	Ph.D.	Research
Comp.Sci.	M.Sc.	Research
Psychology	M.Sc.	Coursework

Calculate

$$ENROLMENT \bowtie_{(Department,Name),(Department,Name)} COURSE$$

2. If the pairs of joining attributes are exactly those that are identically named, we can write

$$ENROLMENT \bowtie COURSE$$

DIVIDE

The DIVISION operation is applied to two Relations

$$R(Z) \div S(X)$$

Where the attributes of R are a subset of the attributes of S.

Let Y be the set of attributes of R that are not attributes of S

R	
A	B
a_1	b_1
a_1	b_2
a_2	b_1
a_3	b_2
a_4	b_1
a_5	b_1
a_5	b_2

S
B
b_1
b_2

Example: $X \subseteq Z$

X = {B}, Z = {A, B}

and Y = Z - X = {A}

DIVIDE

DIVISION is a relation T(Y) that includes a tuple t if tuples t_R appear in R with $t_R[Y] = t$, and with $t_R[X] = t_S$ for every tuple t_S in S.

$$R \div S = \{t : t \times S \subseteq R\}$$

R	
A	B
a_1	$\boxed{b_1}$
a_1	$\boxed{b_2}$
a_2	b_1
a_3	b_2
a_4	b_1
a_5	$\boxed{b_1}$
a_5	$\boxed{b_2}$

S
B
b_1
b_2

Example:

X = {B}, Z = {A, B}, Y = {A}

$t_R[X] = t_S = \{b_1, b_2\}$

In R, there are two satisfied t_R pairs:

$\{a_1b_1, a_1b_2\}$ and $\{a_5b_1, a_5b_2\}$

So $t = t_R[Y] = \{a_1, a_5\}$

T
A
a_1
a_5

DIVIDE

R	
A	B
a_1	b_1
a_1	b_2
a_2	b_1
a_3	b_2
a_4	b_1
a_5	b_1
a_5	b_2

S
B
b_1
b_2

$$R(Z) \div S(X) =$$

T
A
a_1
a_5

DIVIDE

Typical use: which courses are offered by all departments?

$$COURSE \div (\pi_{Department} COURSE)$$

Note: $\{\sigma, \pi, \cup, -, \times\}$ *are sufficient to define all these operations: this is a relationally complete set of operators.*

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R	$\sigma_{<selection\ condition>}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{<attribute\ list>}(R)$
THETA-JOIN	Produces all combinations of tuples from R and S that satisfy the join condition.	$R \bowtie_{<join\ condition>} S$
EQUI-JOIN	Produces all the combinations of tuples from R and S that satisfy a join condition with only equality comparisons.	$R \bowtie_{<join\ condition>} S$
NATURAL-JOIN	Same as EQUIJOIN except that the join attributes of S are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R \bowtie_{<join\ condition>} S$
UNION	Produces a relation that includes all the tuples in R or S or both R and S; R and S must be union compatible.	$R \cup S$
INTERAECTION	Produces a relation that includes all the tuples in both R and S; R and S must be union compatible.	$R \cap S$
DIFFERENCE	Produces a relation that includes all the tuples in R that are not in S; R and S must be union compatible.	R - S
CARTESIAN PRODUCT	Produces a relation that has the attributes of R and S and includes as tuples all possible combinations of tuples from R and S.	$R \times S$
DIVISION	Produces a relation T(X) that includes all tuples t[X] in R(Z) that appear in R in combination with every tuple from S(Y), where Z = X ∪ Y.	$R(Z) \div S(Y)$